A Performance History of Several Multitasking Codes on the NAS Y-MP: 10/30/93 Update

Robert J. Bergeron[1]

Report RND-010 August 1994

NAS Systems Development Branch
NAS Systems Division
NASA Ames Research Center
Mail Stop 258-6
Moffett Field, CA 94035-1000

Abstract

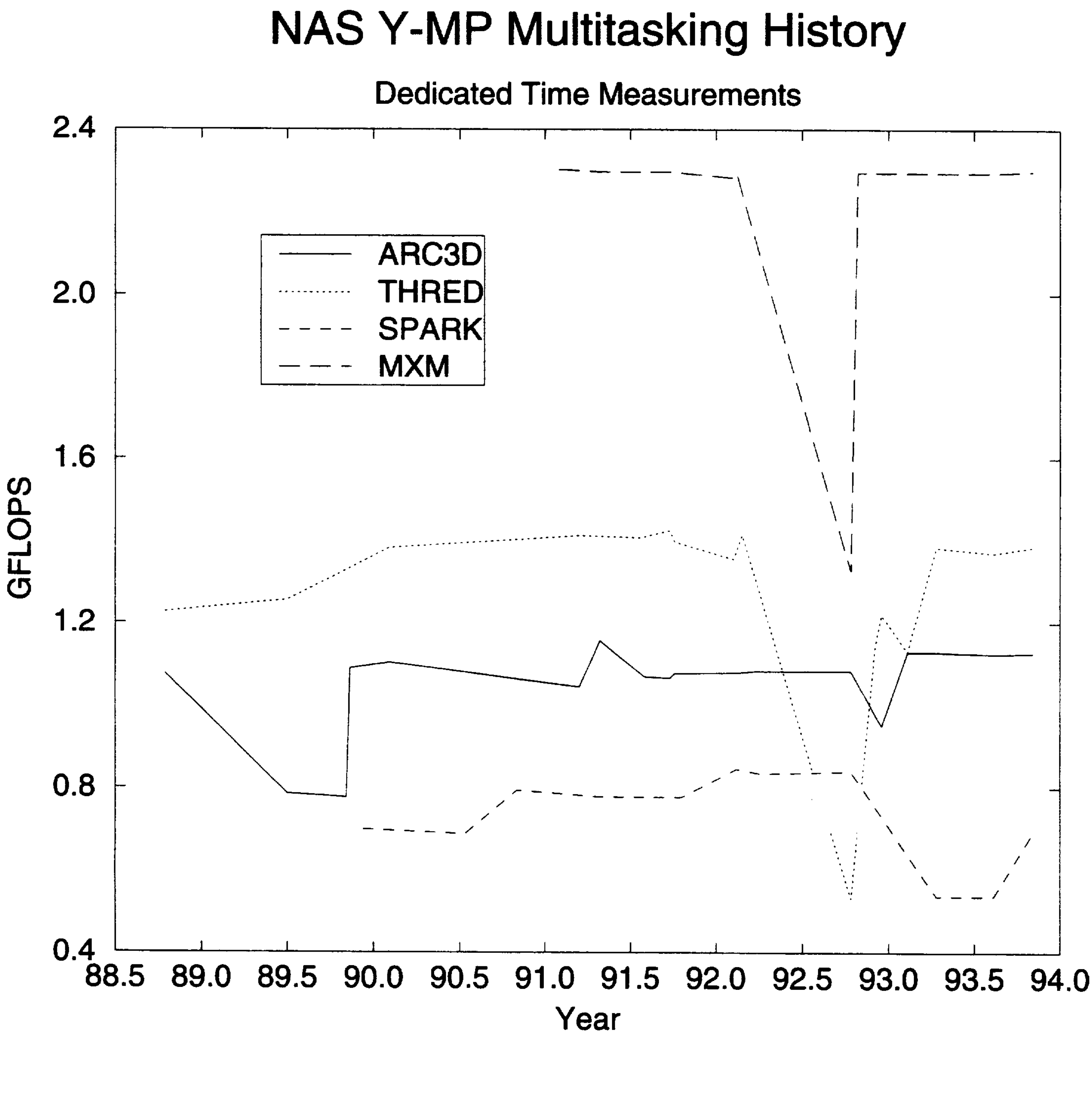
This paper reports performance histories of codes used to ensure that the Y-MP maintains a sustained multitasking capability across the major Cray upgrades. The testing history shows that Cray hardware and software upgrades have a mixed record in maintaining the dedicated time performance of the test codes. While multitasked code performance increased with time, kernel problems retarded execution near the end of the NAS Y-MP service. A Cray library upgrade introduced near the end of service appears responsible for degradation of single processor execution in one code.

1.0 Introduction

This paper describes the performance histories of codes used to ensure that the NAS Cray Y-MP maintained a sustained multitasking capability across the major operating system and compiler upgrades. Previous reports provided performance results for the 1988-1992 period [1,2]. This report details the performance from May 1992 to the retirement of the machine on October 30, 1993.

Two of the test codes, ARC3D and THRED, verified that the Y-MP could sustain 1 GFLOP/sec during the NAS acceptance testing of Y-MP Serial

1. Computer Sciences Corporation, NASA Contract NAS 2-12961, Moffett Field, CA 94035-1000


NAS Y-MP Multitasking History
Dedicated Time Measurements
ARC3D
THRED
SPARK
MXM
GFLOPS
Year
2.4
2.0
1.6
1.2
0.8
0.4
88.5
89.0
89.5
90.0
90.5
91.0
91.5
92.0
92.5
93.0
93.5
94.0

Number (SN) 1002. Periodic upgrades motivated the addition of two other codes to the group. The third code, SPARK, revealed the quality of the parallel code generated by the autotasker and a fourth code, MXM, measured the efficiency of the operating system with respect to autotasking.

All results reported herein reflect calculations performed on a dedicated machine. The figure displays the code performance histories graphically. Dedicated performance monitoring of the Y-MP generally occurred at least once every six months. Addtional monitoring also occurred soon after major compiler or operating system upgrades to establish the effect of system modifications. Test results were communicated informally to Systems Operations and Cray analysts. Tables provide the testing histories; the letter N in the first column denotes new performance data.

In addition to the CFT77 compiler version, the table also provides the versions of the Fortran PreProcessor (FPP) and Fortran MidProcessor (FMP). These utilities enable Fortran codes to exploit parallel processing. FPP parses the user's Fortran source to produce a transformed source file containing parallel directives. FMP converts the parallel directives in the transformed source into Cray intrinsic functions or library calls. The CFT77 compiler can then use the FMP-produced code to create a parallel executable code. The compilers used in this study supported the Fortran 77 standard. A report describing C-90 performance histories will include results for the Fortran 90 compiler.

2.0 ARC3D

Since ARC3D is highly parallel and I/O intensive, its dedicated performance is sensitive to both the UNICOS operating system environment and any local NAS modifications. Before any dedicated time ARC3D execution, a stand-alone test mimicking the ARC3D I/O behavior measures data transfers to and from the SSD. The I/O measurements showed no degradations during the current testing period.

Table 1 shows that ten executions of ARC3D occurred during the current period. Comparison of the 5.0.3.5 compiler listing with the 5.0.1.18 listing indicated comparable vectorization levels, and yet the 10/06/92 test displayed a 5% decrease in performance. A follow-on ARC3D test performed on 11/24/92 failed to achieve the requisite 1 GFLOP performance and led to an instrumented test on 12/07/92.

Table 1: ARC3D Performance History

Date	FPP	FMP	Compiler	UNICOS	Seconds	GFLOPS	Comments
10/13/88	N/A	N/R	3.0	4.0.6+	1114	1.076	1002 Acceptance Test
06/29/89	N/A	N/R	3.0	5.0.8	1525	0.786	UNICOS delays slave process reconnection
11/02/89	N/A	N/R	3.0.1.16	5.0.12	1543	0.777	Improper SSD block size
11/08/89	N/A	N/R	3.0.1.16	5.0.12	1100	1.090	Cray test with SDS
11/08/89	N/A	N/R	3.0.1.16	5.0.12	1164	1.030	SSD + new unit size
02/03/90	N/A	N/R	3.0.2.2	5.0.13	1086	1.104	1030 Upgrade
04/17/90				5.1.8			5.1 Upgrade
11/01/90				6.0.1			6.0 Upgrade
03/13/91	N/A	4.0.3(9)	4.0.3.1	6.0.9	1147	1.045	6.0 File Structure
04/24/91	N/A	5.0(29)	4.0.3.1	6.0.11	1036	1.157	NAS test with SDS (Secondary Data Segments)
07/30/91	N/A	5.0(37)	5.0.0.1	6.0.12	1125	1.065	New cft77 and SRFS (RAM Disk Filesystem /fast)
08/01/91	N/A	5.0(37)	5.0.0.1	6.0.12	1120	1.070	New cft77 and SRFS
08/06/91	N/A	5.0(29)	4.0.3.1	6.0.12	1057	1.134	Old cft77 and SRFS
09/25/91	N/A	5.0(37)	5.0.0.1	6.1.4	1123	1.066	cft77 5.0 and SRFS
10/09/91	N/A	4.0.3(9)	4.0.3.1	6.1.5	1060	1.131	Old cft77 and SRFS
10/09/91	3.03M1	5.0(37)	5.0.0.1	6.1.5	1110	1.078	Microtasker uses Autotasked libraries
02/13/92	3.03M5	5.0.1(6)	5.0.1.1	6.1.5	1108	1.080	256MW Upgrade, MP_DEDICATED=0
03/04/92	3.03N6	5.0.1(10)	5.0.1.18	6.1.5	1104	1.084	256MW Upgrade, MP_DEDICATED=1
N10/06/92	3.03P9	5.0.3(5)	5.0.3.5	6.1.6	1161	1.032	OS Upg- nas.29 pblm-Fair share scheduler?
N11/24/92	3.03P9	5.0.3(5)	5.0.3.5	6.1.6	1315	0.910	Repeat 10/06/92 test
N12/07/92	3.03N6	5.0.1(10)	5.0.1.18	6.1.6	1175	1.020	Try old compiler
N12/07/92	3.03N6	5.0.1(10)	5.0.1.18	6.1.6	1115	1.075	Try old compiler
N12/07/92	3.03N6	5.0.1(10)	5.0.1.18	6.1.6	1255	0.955	Try old compiler
N12/07/92	3.03N6	5.0.1(10)	5.0.1.18	6.1.6	1261	0.951	Try old compiler, MP_DEDICATED=1
N02/09/93	3.03Q6	5.0.4(7)	5.0.3.14	7.0.4.1	1060	1.131	Pre-7.0 installation, MP_DEDICATED=0
N04/06/93	3.03R3	5.0.4(7)	5.0.4.7	7.0.4.1	1061	1.131	New cft and OS Upg
N08/04/93	3.06C1	6.0.0(50)	6.0.0.0	7.0.4.1	1065	1.126	New cft
N10/27/93	3.06C8	6.0.0(50)	6.0.0.4	7.0.4.1	1063	1.128	New cft

N preceding the Date indicates the result obtained in the current period: May 1992-October 1993

N/A-Not Applicable

N/R-Not Recorded

The December test consisted of several ARC3D executions with the most recent version of the compiler that had produced acceptable dedicated-time results. Periodic monitoring of ARC3D execution indicated that the microtasked processes hung near the end of the job. Improper communication between the microtasking library and the kernel resulted in large (as much as 14%) degradations in performance.

The problem was reported to Cray personnel, who forwarded it to the software group at Eagan. While the problem was real, it was reported near the end of the UNICOS 6.1.6 kernel lifetime. A test performed on the NAS Y-MP under a pre-installation version of the UNICOS 7.0.4 kernel gave excellent results and indicated the problem was "historical". Repeated executions of the test under the 7.0.4 kernel at different dates resulted in normal code performance.

A possible cause for the problem was the introduction in May 1992 of automatic HPM monitoring of all user codes. This modification involved a local modification of the user libraries and may have inhibited the proper termination of microtasked processes. The earliest releases of microtasking did display termination problems.

In this period, the ARC3D performance experienced a temporary, severe degradation which was due apparently to a local modification. The subsequent releases of the UNICOS operating system brought ARC3D performance to the high sustained levels measured during earlier releases of the UNICOS kernel.

3.0 THRED

Eight executions of THRED occurred during the testing period. THRED performs file I/O at disk speeds, as opposed to the ARC3D high-speed SSD data transfers.

The performance rate displayed by THRED on 10/06/92 rate was less than the acceptance test requirement. Examination of the executing code with the Cray *crash* utility indicated that the code was unable to exit promptly from a system routine reporting a stack overflow problem. Previous versions of the code had executed correctly despite the overflow problem. Execution with an older version of the UNICOS kernel produced an error abort indicating a stack allocation error.

Although correction of the stack overflow problem led to a THRED execution on 11/24/92 with improved performance, the rate was still 15% less than earlier executions and caused concern.

Table 2: THRED Performance History

Date	FPP	FMP	Compiler	UNICOS	Seconds	GFLOPS	Comments
10/13/88	N/A	N/R	3.0	4.0.6+	1218	1.227	1002 Acceptance
06/29/89	N/A	N/R	3.0	5.0.8	1190	1.255	Minor clock problem
02/03/90	N/A	N/R	3.0.2.2	5.0.13	1080	1.383	1030 Upgrade
04/17/90				5.1.8			5.1 Upgrade
11/01/90				6.0.1			6.0 Upgrade
03/13/91	N/A	4.0.3(9)	4.0.2.1	6.0.9	1057	1.413	6.0 File Structure
07/25/91	N/A	5.0(31)	5.0.0.1	6.0.12	1061	1.408	New cft77
09/25/91	N/A	5.0(37)	5.0.0.1	6.1.4	1049	1.425	UNICOS Upgrade
10/09/91	3.03M1	5.0(37)	5.0.0.1	6.1.5	1069	1.398	Autotasked libraries,MP_DEDICATED=0
02/05/92	3.03M5	5.0.1(6)	5.0.1.1	6.1.5	1102	1.356	256MW Upgrade, MP_DEDICATED=0
02/26/92	3.03N6	5.0.1(10)	5.0.1.18	6.1.5	1056	1.415	MP_DEDICATED=1
N10/06/92	3.03P5	5.0.3(5)	5.0.3.5	6.1.6	2803	0.533	OS Upg-possible Fortran stack overflow pblm
N10/06/92	3.03P5	5.0.3(5)	5.0.3.5	6.1.5	N/A	N/A	6.1.5 kernel- abort w/ "bstack allocation error"
N11/24/92	3.03P5	5.0.3(5)	5.0.3.5	6.1.6	1310	1.141	Fortran stack overflow resolved-performance poor
N12/07/92	3.03N6	5.0.1(10)	5.0.1.18	6.1.6	1213	1.222	Try old compiler
N12/07/92	3.03N6	5.0.1(10)	5.0.1.18	6.1.6	1130	1.323	Try old compiler
N04/06/93	3.03R3	5.0.4(7)	5.0.4.7	7.0.4.1	1080	1.383	New cft and OS Upg
N08/04/93	3.06C1	6.0.0(50)	6.0.0.0	7.0.4.1	1090	1.371	New cft
N10/27/93	3.06C8	6.0.0(50)	6.0.0.4	7.0.4.1	1078	1.386	New cft

N preceding the Date indicates the result obtained in the current period: May 1992-October 1993

N/A-Not Applicable

N/R-Not Recorded

Two THRED executions on 12/07/92 also gave reduced performance. The subsequent release of the UNICOS 7.0.4 kernel brought THRED performance to the previous levels.

In this period, THRED performance experienced a temporary, severe degradation which was apparently due to a local library modification.

4.0 SPARK

Seven dedicated executions of the SPARK code occurred during this period and Figure 1 shows that a substantial performance decrease occurred in mid-92. The performance of SPARK in singletasked mode decreased from 185 MFLOPS to 175 MFLOPS during this period, and this decrease led to a degradation in parallel performance since FPP uses the same Fortran source as input.

The singletasked reduction occurred in production as well as in dedicated time. Comparison of the compiler listings with previous listings indicated no difference in vectorization. Execution with previous releases of the compiler under the 7.0.4 kernel also displayed the singletasked performance reduction. The evidence indicated that a library change may have been responsible. This conclusion could have been checked by execution of the 6.1.5 version of SPARK, but it was not available.

Measurement of the CPU time spent by SPARK in scientific library routines indicated that the code spent about 50% of its CPU time in these routines. (The placement of timing code around the library calls was highly intrusive and hindered vectorization of the code; a less intrusive approach would count the total number of library calls and apply an average time per call). SPARK spends a significant amount of CPU time in the scientific library because it models finite-rate chemistry and makes unusually strong use of the exponential function. Most NAS codes do not model rate processes and thus would not suffer the SPARK performance degradation.

Table 3: SPARK Performance History

Date	FPP	FMP	Compiler	UNICOS	Seconds	GFLOPS	Comments
5.1 Option							
12/14/89	2.26B18	3.1(33)	3.0.2.2	5.0.13	18.18	0.700	12.73 billion floating point operations
12/14/89	2.24S6		3.0.2.2	5.0.13	15.37	0.828	Old Libraries
07/13/90	2.24S6		3.0.2.2	5.1.10	14.49	0.879	
07/13/90	3.00Z51	4.0.2(12)	4.0.	5.1.10	18.48	0.689	
11/01/90				6.0.1			6.0 Upgrade
11/06/90	3.00Z36	4.0.1(38)	4.0.1	6.0.11	16.01	0.795	
04/25/91	3.00Z61	4.0.3(9)	4.0.3.1	6.0.11	151.1	0.732	110.67 billion floating point operations
04/25/91	3.03Y4	5.0(29)	5X402417	6.0.11	142.2	0.778	
10/23/91	3.03M5	5.0.1(6)	5.0.0.0	6.1.5	142.4	0.777	UNICOS Upgrade
02/13/92	3.03M5	5.0.1(6)	5.0.0.1	6.1.5	130.7	0.846	256MW Upgrade
03/04/92	3.03N6	5.0.1(10)	5.0.0.18	6.1.5	132.3	0.836	Compiler Upgrade
N10/06/92	5.03P9	5.0.3(5)	5.0.3.5	6.1.6	131.6	0.840	Compiler Upgrade
N04/06/93	3.03R3	5.0.4(7)	5.0.4.7	7.0.4.1	205.6	0.538	New cft and OS Upg
N08/04/93	3.06C1	6.0.0(50)	6.0.0.0	7.0.4.1	205.6	0.538	New cft
N10/27/93	3.06C8	6.0.0(50)	6.0.0.4	7.0.4.1	159.2	0.695	New cft
6.0 Option							
04/25/91	3.00Z61	4.0.3(9)	4.0.3.1	6.0.11	150.8	0.734	
04/25/91	3.03Y4	5.0(29)	5X402417	6.0.11	143.7	0.770	
N04/06/93	3.03R3	5.0.4(7)	5.0.4.7	7.0.4.1	229.6	0.481	New cft and OS Upg
N08/04/93	3.06C1	6.0.0(50)	6.0.0.0	7.0.4.1	229.0	0.483	New cft
N10/27/93	3.06C8	6.0.0(50)	6.0.0.4	7.0.4.1	177.3	0.623	New cft

N preceding the Date indicates the result obtained in the current period: May 1992-October 1993

Execution on 10/27/93 gave an increased performance which was attributed to improvements in FPP because the singletasked performance of the code remained poor.

5.0 MXM

The compiler upgrade to version 5.0 of CFT77 includes a default which allows the autotasker to (silently) replace the autotasked matrix multiply with the SCILIB call SGEMMX. Since MXM is always compiled with default autotasking options, such replacement frustrates the purpose of the test. While an FPP option will disable this default, it is good practice to check the output of the preprocessor before executing this test.

Table 4 shows that five tests were conducted during the April 1991-April 1992 period.The test on 10/06/92 required 100% more CPU time for successful execution of the parallel version and this was attributed to the same cause as the ARC3D problem. Although a repeat of this test on 10/20/92 gave acceptable performance, the 7-CPU version of MXM was inexplicably dumping core at the end of its successful execution. Subsequent executions with newer UNICOS versions obtained acceptable performance rates with no core dumps.

Table 4: MXM Performance History

Date	FPP	FMP	Compiler	UNICOS	Seconds	GFLOPS	Comments
01/30/91	2.26B18	3.1(33)	3.0.2.2	6.0.9	0.119	2.278	n=512
01/30/91	2.24S6	——	3.0.2.2	6.0.9	59.79	2.302	n=4096
02/13/91	3.00Z61	4.0.3(9)	4.0.3.1	6.0.11	59.79	2.302	n=4096
04/28/91	3.03Y4	5.0(29)	5X402417	6.0.11	59.91	2.297	n=4096
09/25/91	3.03M1	5.0(29)	5.0.0.1	6.1.4	59.91	2.297	n=4096
02/05/92	3.03M1	5.0.1(6)	5.0.1.1	6.1.5	60.24	2.284	CPU error
02/13/92	3.03M1	5.0.1(6)	5.0.1.1	6.1.5	59.93	2.296	256MW Upgrade
N10/06/92	3.03P5	5.0.3(5)	5.0.3.5	6.1.6	103.70	1.327	OS Upg nas.29 pblm;NCPU=7 dump core
N10/20/92	3.03P5	5.0.3(5)	5.0.3.5	6.1.6	59.95	2.296	OS Upg-repeat test;NCPU=7 dump core
N04/06/93	3.03R3	5.0.4(7)	5.0.4.7	7.0.4.1	59.98	2.295	OS Upg
N08/04/93	3.06C1	6.0.0(50)	6.0.0.0	7.0.4.1	59.85	2.230	New cft
N10/27/93	3.06C8	6.0.0(50)	6.0.0.4	7.0.4.1	59.86	2.299	Final

N preceding the Date indicates the result obtained in the current period: May 1992-October 1993

Table 5 provides MXM efficiencies measured for 2 through 8 CPUs and a 16-CPU efficiency extrapolated from a linear curve fit to the measured data. This table shows that, with the exception of the 6.1.6 error, Cray upgrades maintained the performance of autotasked matrix multiply during this period. Figure 1 shows that performance of MXM has been constant throughout the testing history.

Table 5: MXM Efficiency History

Date	CPUS							
	1	2	3	4	5	6	7	8	Comments
									Comments
01/30/91	1.000	0.999	0.992	0.991	0.991	0.988	0.983	0.989	E=0.969 for 16 CPUs
02/13/91	1.000	0.994	0.994	0.997	0.996	0.992	0.991	0.987	E=0.978 for 16 CPUs
09/25/91	1.000	0.990	0.986	0.987	0.986	0.986	0.984	0.981	E=0.965 for 16 CPUs
02/05/92	1.000	1.002	1.116	1.116	1.113	1.113	1.110	1.101	CPU error
02/13/92	1.000	0.993	1.008	1.005	1.005	1.003	1.002	0.999	CPU release error
N10/06/92	1.000	1.001	1.001	0.995	0.998	0.996	0.994	0.511	6.1.6 kernel problems
N10/20/92	1.000	1.001	1.001	0.995	0.998	0.996	0.994	0.992	E=0.984 for 16 CPUs
N04/06/93	1.000	0.998	0.998	1.006	1.003	1.002	0.999	0.997	E=0.983 for 16CPUs
N08/04/93	1.000	0.988	0.977	1.007	0.990	0.976	0.918	0.990	E=0.983 for16 CPUs
N10/27/93	1.000	0.988	0.977	1.007	0.990	0.976	0.918	0.990	E=0.983 for 16 CPUs

N preceding the Date indicates the result obtained in the current period: May 1992-October 1993

6.0 Conclusions

Measurements of multitasked performance during dedicated time provide an exquisite measure of the system state. Measurements made throughout the service lifetime of the NAS Y-MP indicate that Cray operating system and compiler upgrades have had a mixed effect upon the performance of the test multitasking codes. The performance histories show that the dedicated time testing is a valuable exercise and should continue on future HSPs.

For the simplest code, the matrix multiply (MXM), upgrades maintained performance. Codes employing microtasking, ARC3D and THRED, displayed 5 and 12% performance increases as the operating system and compiler matured. However, testing of these codes uncovered several problems in the releases of the UNICOS operating system with some releases reducing performance by factors of 2 to 3.

The SPARK code performance was 8% slower at the end of the Y-MP lifetime than it was at the beginning. The performance decrease appeared to be a slowdown in the library routines and this decrease was observed in normal production as well as dedicated time.

7.0 Acknowledgment

Thanks to Bob Ciotti and Larry Davitt for reviewing this paper.

8.0 References

[1] R.J. Bergeron. A Performance History of Several Multitasking Codes on the NAS Y-MP. NAS Report RND-91-008, July 1991.

[2] R.J. Bergeron. A Performance History of Several Multitasking Codes on the NAS Y-MP: 4/15/92 Update. NAS Report RND-92-019, July 1991.

RND TECHNICAL REPORT

Title: A Performance History of Several Multitasking Codes on the NAS Y-MP:10/30/93 Update

Author(s):
Robert J. Bergeron

Reviewers:
"I have carefully and thoroughly reviewed this technical report. I have worked with the author(s) to ensure clarity of presentation and technical accuracy. I take personal responsibility for the quality of this document."

Signed: Larry Davitt

Name: LARRY DAVITT

Signed: Bob Ciotti

Name: Bob Ciotti

Branch Chief:

Approved: for BTB
John L Kaufman

Date & TR Number:

August 1994 RND-94-010